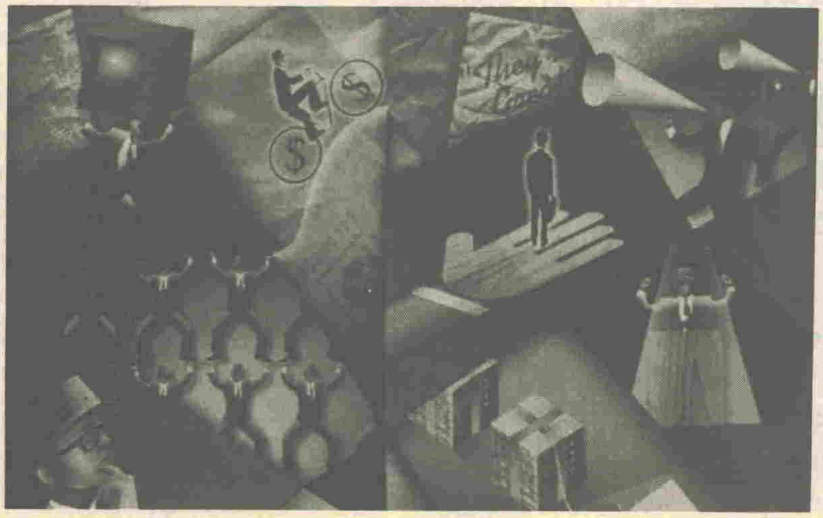
สุทธิศักดิ์ พงศ์ธนาพาณิช

SOFT MAIL

เปิดโลกมัลติมีเดียด้วย VB/Win ตอน จัดการกับ Palette ของวินโดวส์ 95 ด้วย VB/Win #2



เจาะลึกถึงเทคนิคการแก้ไขสี ในถาดตรรกะและถาดสีระบบ

จากตอนที่แล้วผู้เขียนได้อธิบายถึงพื้นฐานการจัดการ palette ของวินโดวส์ 95 และความแตกต่างระหว่างถาดสีตรรกะ (logical palette) และถาดสีระบบ (physical palette) และได้แสดงแอพพลิเคชันตัวอย่างการสร้างถาดสี (palette plate) สำหรับแสดงผล 256 สี และการใช้ฟังก์ชัน RGB ของ VB/Win ในการปรับส่วนผสมระหว่างแม่สีทั้งสามสี ซึ่งโดยตัวของ VB/Win แล้ว ไม่สนับสนุนการจัดการกับ palette โดยตรง แต่จะมีการเพิ่มความสามารถในการแสดงผลสีต่าง ๆ จากถาดสีตรรกะให้กับฟังก์ชันด้านกราฟิก เช่น Line, Circle หรือ Pset เป็นต้น ดังนั้นถ้าหากผู้อ่านได้นำเทคนิคการจัดการ palette ด้วยฟังก์ชันวินโดว์ API ไปใช้แล้ว ก็จะช่วยขยายขีดความสามารถของแอพพลิเคชันที่สร้างโดย VB/Win ได้อย่างมากทีเดียว

สำหรับในบทความตอนนี้ก็จะเป็นเนื้อหาที่ต่อเนื่องจากตอนที่แล้ว แต่บทความตอนนี้จะเจาะลึกลงไปถึงเทคนิคการแก้ไขสีในถาดสีตรรกะและถาดสีระบบ ซึ่งเทคนิคนี้ได้ถูกนำไปใช้ในส่วนของการปรับแต่งสี (Color customization) โดย Control Panel ของวินโดวส์ โดยอาศัยเทคนิคในบทความตอนนี้ผู้อ่านก็จะสามารถสร้างสีโดยมีส่วนผสมของแม่สีหลักต่าง ๆ ได้ตามที่ต้องการ และนอกจากนี้ในโปรแกรมตัวอย่างผู้เขียนยังจะแนะนำวิธีการสร้างถาดสีตรรกะ โดยไม่จำเป็นต้องใช้ palette ที่มากับไฟล์ .DIB อีกด้วย

เนื่องจากวินโดวส์เป็นระบบปฏิบัติการที่ทำงานแบบมัลติทาสกิง ดังนั้นจึงมีเทคนิคการจัดการ palette ที่ค่อนข้างซับซ้อน และมีการ mapping สีระหว่างถาดสีตรรกะกับถาดสีระบบ สำหรับแอพพลิเคชันที่อยู่ในสถานะแอกทีฟอย่างต่อเนื่อง และเพื่อให้ผู้อ่านได้เข้าใจถึงความแตกต่างระหว่างการเปลี่ยนแปลงสีในถาดสีตรรกะและถาดสีระบบ ในการรันโปรแกรมตัวอย่างขอให้ผู้อ่านสังเกตการเปลี่ยนแปลงสีของรูปภาพใน desktop ของวินโดวส์ เพื่อเปรียบเทียบความแตกต่างที่เกิดขึ้น

ถาดสีตรรกะและถาดสีระบบ

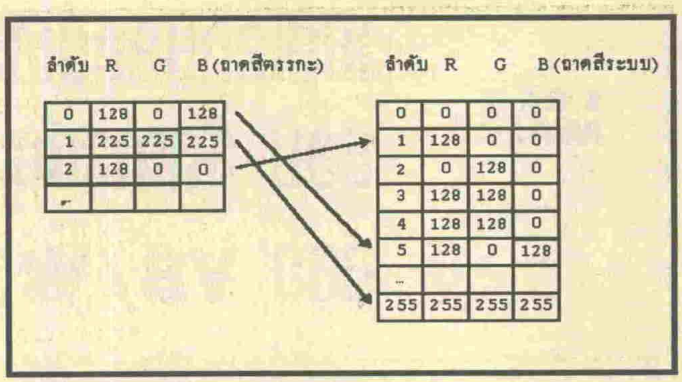
จากบทความตอนที่แล้ว ผู้เขียนได้อธิบายถึง palette manager ของวินโดวส์ 95 ไปแล้ว ดังนั้นในบทความตอนนี้ก็จะขอทำความเข้าใจเพิ่มเติมเกี่ยวกับข้อแตกต่างระหว่างถาดสีตรรกะและถาดสีระบบ แอพพลิเคชันหนึ่ง ๆ ที่ทำงานจะมีการอ้างถึงถาดสีของตนเองเป็นการเฉพาะ โดยที่เป็นถาดสีที่ถูกสร้างขึ้นมาและจัดเก็บในหน่วยความจำ ซึ่งเราเรียกถาดสีชนิดนี้ว่า ถาดสีตรรกะ ดังนั้นวินโดวส์จึงสามารถที่จะสร้างถาดสีตรรกะขึ้นมาได้อย่างไม่จำกัด แต่สีที่จะถูกใช้ในการแสดงผลหน้าต่าง

หรือ desktop นั้น ก็จะเป็นถาดสีที่ถูกจัดเก็บในฮาร์ดแวร์ และสำหรับระบบหนึ่ง ๆ จะสามารถมีถาดสีนี้ได้เพียงหนึ่งเดียวเท่านั้น ซึ่งเราเรียกถาดสีชนิดนี้ว่า ถาดสีระบบ

ในขณะที่แอพพลิเคชันหนึ่งมีความต้องการที่จะใช้ถาดสีของตนเอง แอพพลิเคชันก็จะต้องส่งข้อความ (windows message) ไปยังวินโดวส์ เพื่อที่จะให้วินโดวส์ทำการเลือกถาดสีตรรกะของแอพพลิเคชันนั้น ๆ ลงใน device context และกำหนดให้ palette manager ทำการโหลดถาดสีตรรกะใน device context นี้ลงในถาดสีระบบ ซึ่งเราเรียกการทำงานในส่วนนี้ว่า realizing

จากรูปที่ 1 ในการ realize ถาดสีนั้น palette manager จะจับคู่ระหว่างค่าของสี Red, Green และ Blue ที่ตรงกันของถาดสีตรรกะและถาดสีระบบ โดยไม่สนใจลำดับของสี (คอลัมน์แรก) และในการ realize สีแต่ละครั้ง palette manager จะไม่โหลดสีทั้งหมด 256 เพียงครั้งเดียว แต่ palette manager จะโหลดเฉพาะสีที่ปรากฏในถาดสีตรรกะเท่านั้น และถ้าหากหลาย ๆ ถาดสีตรรกะที่ถูกโหลด วินโดวส์จะกำหนดให้มีการ realize ได้ไม่เกิน 256 สี โดยในความเป็นจริงวินโดวส์จะทำการ activate สีเพียง 236 เท่านั้น ทั้งนี้เนื่องจากวินโดวส์จะกัน 20 สีเอาไว้สำหรับใช้แสดงผลส่วนต่าง ๆ ของวินโดวส์ ซึ่งเราเรียกสีทั้ง 20 สีนี้ว่า static color และในเอกสารของไมโครซอฟท์ได้ระบุว่า ไม่แนะนำโปรแกรมเมอร์ทำการแก้ไข static color

แต่สำหรับฟังก์ชันวินโดว์ API ด้าน GDI นั้นจะมีวิธีการจับคู่สีในถาดสีตรรกะและถาดสีระบบที่แตกต่างจาก palette manager โดยที่ palette manager จะทำการสร้างตารางที่เรียกว่า palette mapping ซึ่งจะระบุลำดับของสี (color index) ที่จะให้สีที่ตรงกันระหว่างถาดสีตรรกะและถาดสีระบบ ดูตัวอย่างได้จากรูปที่ 3 ในบทความตอนที่แล้ว



รูปที่ 1 แสดงขั้นตอนการ realize ถาดสี

การเปลี่ยนสีในถาดสี

ตามที่กล่าวมาแล้วข้างต้น บทความตอนนี้จะเน้นเฉพาะวิธีการเปลี่ยนสีในถาดสีเท่านั้น โดยจะแนะนำทั้งเทคนิควิธีการเปลี่ยนสีในถาดสีตรรกะและถาดสีระบบ ซึ่งวิธีทั้งสองจะส่งผลที่แตกต่างกันซึ่งผู้อ่านจะสังเกตได้จากการทดลองใช้งานโปรแกรมตัวอย่าง แต่ก่อนที่ผู้เขียนจะอธิบายถึงเทคนิคการเปลี่ยนสีในถาดสี ก็จะขออธิบายถึงรายละเอียดของฟังก์ชันวินโดว์ API ที่เกี่ยวข้องเสียก่อน เพื่อให้ผู้อ่านสามารถเข้าใจโค้ดในส่วนที่ทำการเปลี่ยนสีในถาดสีได้ง่ายขึ้น เนื่องจากมีฟังก์ชันวินโดว์ API บางส่วนที่ผู้เขียนได้อธิบายไปแล้วในบทความตอนที่ผ่านมา เช่น DeleteObject, SeleteObject เป็นต้น ดังนั้นจึงไม่ขออธิบายซ้ำในบทความตอนนี้ จึงขอให้ผู้อ่านที่สนใจได้ศึกษาเพิ่มเติมจากบทความตอนที่แล้วครับ

CreatePalette

การประกาศ
```
Private Declare Function CreatePalette Lib _
"gdi32" (lpLogPalette As LOGPALETTE) As Long
```

หน้าที่ สร้างถาดสีตรรกะตามข้อมูลที่กำหนด

พารามิเตอร์
lpLogPalette ข้อมูลโครงสร้างชนิด LOGPALETTE เกี่ยวกับสีที่ใช้ในการสร้างถาดสีตรรกะ

ตัวอย่างการใช้
```
LogicalPalette.palVersion = &H300
LogicalPalette.palNumEntries = 236
            LogicalPalette.palPalEntry(ClrIdx).peRed = 128
LogicalPalette.palPalEntry(ClrIdx).peGreen = 0
LogicalPalette.palPalEntry(ClrIdx).peBlue = 255
LogicalPalette.palPalEntry(ClrIdx).peFlags = PC_RESERVED
            hCurrentPalette = CreatePalette(LogicalPalette)
```

ค่าที่ส่งกลับ
= 0 ถ้าหากฟังก์ชันทำงานล้มเหลว
<> 0 หมายเลข handle ของถาดสีตรรกะที่ถูกสร้าง

หมายเหตุ เมื่อมีการสร้างถาดสีตรรกะเรียบร้อยแล้ว ผู้อ่านต้องเขียนโค้ดเพื่อทำการเลือกถาดสีตรรกะที่ถูกสร้างลงใน device context โดยใช้ฟังก์ชัน SelectPalette จากนั้นจึงทำการ realize ด้วยฟังก์ชัน RealizePalette อีกครั้ง และถ้าหากผู้อ่านไม่ต้องการใช้งานถาดสีตรรกะอีกต่อไป ก็ควรที่จะลบทิ้งเสีย เพื่อเป็นการลดการใช้ทรัพยากรของระบบด้วยฟังก์ชัน DeleteObject สำหรับข้อมูลชนิดโครงสร้าง LOGPALETTE นั้นประกอบด้วยสมาชิก 3 ตัวดังนี้

→

palVersion	ข้อมูลชนิดเลขจำนวนเต็ม กำหนดเวอร์ชันของวินโดวส์ (&H300)
palNumEntries	ข้อมูลชนิดเลขจำนวนเต็ม กำหนดจำนวนรายการสีในถาดสีตรรกะ
palPalEntry	ข้อมูลชนิด PALETTEENTRY กำหนดรายละเอียดของแม่สี ที่ใช้ในการสร้างสีต่าง ๆ ของถาดสีตรรกะ ซึ่งสมาชิกของโครงสร้าง PALETTEENTRY มีดังต่อไปนี้
peRed	ข้อมูลชนิดไบต์ กำหนดความเข้มของแม่สีแดง (0 ถึง 255)
peGreen	ข้อมูลชนิดไบต์ กำหนดความเข้มของแม่สีเขียว (0 ถึง 255)
peBlue	ข้อมูลชนิดไบต์ กำหนดความเข้มของแม่สีน้ำเงิน (0 ถึง 255)
peFlags	ข้อมูลชนิดไบต์ กำหนดวิธีการใช้งานแต่ละรายการในถาดสีตรรกะ โดยที่ค่าคงที่ 1 (PC_RESERVED) หมายถึง กำหนดให้ถาดสีตรรกะที่ถูกสร้างสำหรับการทำ animation ดังนั้นจึงทำให้รายการสีของถาดสีมีการเปลี่ยนแปลงค่าบ่อย ๆ

■

GetPaletteEntries

การประกาศ
```
Private Declare Function GetPaletteEntries _
Lib "gdi32" (ByVal hPalette As Long, ByVal _
wStartIndex As Long, ByVal wNumEntries As Long, _
lpPaletteEntries As PALETTEENTRY) As Long
```

หน้าที่ อ่านข้อมูลของรายการสีในถาดสีตรรกะ ตามจำนวนรายการที่ถูกกำหนด

พารามิเตอร์

hPalette	ข้อมูลชนิดเลขจำนวนเต็ม long กำหนดหมายเลข handle ของถาดสีตรรกะ
wStartIndex	ข้อมูลชนิดเลขจำนวนเต็ม long กำหนดรายการแรกของถาดสีตรรกะที่ต้องการอ่านข้อมูล
wNumEntries	ข้อมูลชนิดเลขจำนวนเต็ม long กำหนดจำนวนรายการสีที่ต้องการ
lpPaletteEntries	ข้อมูลชนิดโครงสร้าง PALETTEENTRY

ตัวอย่างการใช้
```
' retrive color value from palette.
    Dummy = GetPaletteEntries(hCurrentPalette, _
                SelectColorIdx, 1, CurrentPaletteEntry)

    ColorGradient(0).Value = 255 - CurrentPaletteEntry.peRed
    ColorGradient(1).Value = 255 - CurrentPaletteEntry.peGreen
    ColorGradient(2).Value = 255 - CurrentPaletteEntry.peBlue
```

ค่าที่ส่งกลับ
= 0 ถ้าหากฟังก์ชันทำงานล้มเหลว
<> 0 ถ้าหากฟังก์ชันทำงานสำเร็จ ซึ่งหมายถึงจำนวนรายการสีที่ถูกอ่านข้อมูล

หมายเหตุ สำหรับจำนวนรายการสีที่ต้องการอ่านข้อมูลนั้น ต้องไม่มากกว่าจำนวนรายการสีที่เหลือในถาดสี โดยนับเริ่มต้นจากลำดับสีที่ถูกกำหนดให้กับพารามิเตอร์ wStartIndex

RealizePalette

การประกาศ
```
Private Declare Function RealizePalette _
Lib "gdi32" (ByVal hDC As Long) As Long
```

หน้าที่ ทำการจับคู่รายการสีของถาดสีตรรกะกับถาดสีระบบ

พารามิเตอร์ hDC ข้อมูลชนิดเลขจำนวนเต็ม long กำหนดหมายเลข handle ของ device context ที่ถาดสีตรรกะถูกเลือกไว้ภายใน

ตัวอย่างการใช้
```
hSystemPalette = SelectPalette(Picture1.hDC, hCurrentPalette, False)
Dummy = RealizePalette(Picture1.hDC)
```

ค่าที่ส่งกลับ
= GDI_ERROR ถ้าหากฟังก์ชันทำงานล้มเหลว
> 0 ถ้าหากฟังก์ชันทำงานสำเร็จ และรายงานจำนวนรายการสีที่ถูกจับคู่กับถาดสีระบบ

หมายเหตุ ฟังก์ชันนี้จะทำการแก้ไขถาดสีในดีไวซ์ ตามที่ถูกกำหนดให้กับฟังก์ชัน โดยถ้าหากดีไวซ์เป็นชนิด memory DC ตารางสีสำหรับบิตแม็พที่ถูกเลือกลงใน DC ก็จะถูกแก้ไข แต่ถ้าหากดีไวซ์เป็นชนิด display DC ถาดสีระบบก็จะถูกแก้ไข จากตัวอย่างข้างต้น เมื่อทำการเลือกถาดสีตรรกะลงใน DC ของคอนโทรล Picture Box แล้ว โค้ดบรรทัดถัดไปก็ทำการ realize ถาดสีใน DC ดังกล่าว

เปิดโลกมัลติมีเดียด้วย VB/Win
ตอน จัดการกับ Palette ของวินโดวส์ 95 ด้วย VB/Win #2

SelectPalette

การประกาศ Private Declare Function SelectPalette Lib "gdi32" _
(ByVal hDC As Long, ByVal hPalette As Long, _
ByVal bForceBackground As Long) As Long

หน้าที่ เลือกถาดสีตรรกะลงใน device context

พารามิเตอร์

hDC	ข้อมูลชนิดเลขจำนวนเต็ม long กำหนดหมายเลข handle ของ device context ที่ถาดสีตรรกะถูกเลือกไว้ภายใน
hPalette	ข้อมูลชนิดเลขจำนวนเต็ม long กำหนดหมายเลข handle ของถาดสีตรรกะ ที่ต้องการโหลดลงใน device context
bForceBackground	ข้อมูลชนิดเลขจำนวนเต็ม long กำหนดถาดสีตรรกะที่ถูกโหลดลงใน device context จะเป็นถาดสี background หรือถาดสี foreground

ตัวอย่างการใช้

```
hSystemPalette = SelectPalette(Picture1.hDC, hCurrentPalette, False)
Dummy = RealizePalette(Picture1.hDC)
```

ค่าที่ส่งกลับ

= 0	ถ้าหากฟังก์ชันทำงานล้มเหลว
<> 0	ถ้าหากฟังก์ชันทำงานสำเร็จ และรายงานหมายเลข handle ของถาดสีก่อนหน้านี้ใน device context

หมายเหตุ สำหรับค่าของ bForceBackground ผู้อ่านสามารถกำหนดได้ ดังนี้

True	กำหนดให้ถาดสีตรรกะถูกจับคู่สีในถาดสีระบบด้วยวิธีที่ดีที่สุด
False	กำหนดให้มีการคัดลอกถาดสีตรรกะลงใน display DC เมื่อแอพพลิเคชันอยู่ในสถานะ active และได้รับโฟกัส ในการเลือกถาดสีตรรกะลงใน device context นั้น ผู้อ่านสามารถเลือกลงได้สำหรับหลาย ๆ device context ซึ่งเมื่อมีการแก้ไขถาดสีตรรกะ ก็จะส่งผลต่อ device context ที่บรรจุถาดสีตรรกะทันที

SetPaletteEntries

การประกาศ Private Declare Function SetPaletteEntries _
Lib "gdi32" (ByVal hPalette As Long, ByVal _
wStartIndex As Long, ByVal wNumEntries As Long, _
lpPaletteEntries As PALETTEENTRY) As Long

หน้าที่ กำหนดค่าของสีแดง เขียว และน้ำเงิน และข้อมูลที่เกี่ยวข้องในถาดสีตรรกะ ตามจำนวนรายการสีที่ถูกกำหนด

พารามิเตอร์

hPalette	ข้อมูลชนิดเลขจำนวนเต็ม long กำหนดหมายเลข handle ของถาดสีตรรกะที่ต้องการโหลดลงใน device context
wStartIndex	ข้อมูลชนิดเลขจำนวนเต็ม long กำหนดรายการแรกของถาดสีตรรกะที่ต้องการกำหนดข้อมูล
wNumEntries	ข้อมูลชนิดเลขจำนวนเต็ม long กำหนดจำนวนรายการสีที่ต้องการ
lpPaletteEntries	ข้อมูลชนิดโครงสร้าง PALETTEENTRY

ตัวอย่างการใช้

```
hSystemPalette = GetStockObject(DEFAULT_PALETTE)
hCurrentPalette = SelectPalette(Picture1.hDC, hSystemPalette, False)
Dummy = SetPaletteEntries(hCurrentPalette, SelectColorIdx, 1, NewPaletteEntry)
hSystemPalette = SelectPalette(Picture1.hDC, hCurrentPalette, False)
Dummy = RealizePalette(Picture1.hDC)
```

ค่าที่ส่งกลับ

= 0	ถ้าหากฟังก์ชันทำงานล้มเหลว
<> 0	ถ้าหากฟังก์ชันทำงานสำเร็จ ซึ่งหมายถึงจำนวนรายการสีที่ถูกกำหนดข้อมูลโดยฟังก์ชันนี้

หมายเหตุ ฟังก์ชันนี้จะทำการกำหนดค่าของสีแดง เขียว และน้ำเงิน และข้อมูลที่เกี่ยวข้องลงในถาดสีตรรกะเท่านั้น ดังนั้นจึงไม่มีผลต่อถาดสีระบบ

เปิดโลกมัลติมีเดียด้วย VB/Win

ตอน จัดการกับ Palette ของวินโดวส์ 95 ด้วย VB/Win #2

AnimatePalette

การประกาศ	Private Declare Sub AnimatePalette Lib "gdi32" (ByVal _	
	hPalette As Long, ByVal wStartIndex As Long, _	
	ByVal wNumEntries As Long, lpPaletteColors As PALETTEENTRY)	
หน้าที่	กำหนดข้อมูลของแม่สีและข้อมูลที่เกี่ยวข้องลงในถาดสีตรรกะ ตามจำนวนรายการสีที่ถูกกำหนด	
พารามิเตอร์	hPalette	ข้อมูลชนิดเลขจำนวนเต็ม long กำหนดหมายเลข handle ของถาดสีตรรกะ ที่ต้องการโหลดลงใน device context
	wStartIndex	ข้อมูลชนิดเลขจำนวนเต็ม long กำหนดรายการแรกของถาดสีตรรกะที่ต้องการกำหนดข้อมูล
	wNumEntries	ข้อมูลชนิดเลขจำนวนเต็ม long กำหนดจำนวนรายการสีที่ต้องการ
	lpPaletteEntries	ข้อมูลชนิดโครงสร้าง PALETTEENTRY

ตัวอย่างการใช้

```
NewPaletteEntry.peRed = 255 - ColorGradient(0).Value
NewPaletteEntry.peGreen = 255 - ColorGradient(1).Value
NewPaletteEntry.peBlue = 255 - ColorGradient(2).Value
NewPaletteEntry.peFlags = PC_RESERVED
AnimatePalette hCurrentPalette, SelectColorIdx, 1, NewPaletteEntry
```

ค่าที่ส่งกลับ ไม่มี

หมายเหตุ ฟังก์ชันนี้จะสามารถแก้ไขค่าของสีได้เฉพาะกับรายการที่แฟลกมีค่าเท่ากับ PC_RESERVED เท่านั้น

สำหรับโค้ดในการสร้างถาดสีตรรกะนั้นขอให้ผู้อ่านดูได้จากการทำงานของโปรแกรม LogPal ในหัวข้อถัดไป ส่วนวิธีการเปลี่ยนสีในถาดสีทั้งสองประเภทนั้นมีเทคนิคง่าย ๆ ดังนี้

1. สร้างตัวแปรชนิดข้อมูล PALETTEENTRY สำหรับจัดเก็บค่าสีและแฟลกของสีที่ต้องการแก้ไข โดยผู้อ่านสามารถสร้างเป็นตัวแปรอะเรย์ได้ ถ้าหากจำนวนรายการสีที่ต้องการแก้ไขมากกว่า 1 รายการ

```
Dim NewPaletteEntry As PALETTEENTRY
```

2. จัดเก็บค่าตัวเลขสีและแฟลกของสีที่ต้องการแก้ไขลงในตัวแปร NewPaletteEntry โดยที่ค่าแม่สีทั้งสาม (peRed, peGreen และ peBlue) สามารถมีค่าได้ตั้งแต่ 0 ถึง 255 ส่วนค่าของแฟลก (peFlags) ก็ขึ้นกับชนิดของถาดสีที่ถูกแก้ไข ดังนี้ การแก้ไขถาดสีตรรกะค่าของแฟลกต้องเท่ากับ 0 ส่วนการแก้ไขถาดสีระบบค่าของแฟลกต้องเท่ากับ 1 (PC_RESERVED)

```
NewPaletteEntry.peRed = 255 -
ColorGradient(0).Value
NewPaletteEntry.peGreen = 255 -
ColorGradient(1).Value
NewPaletteEntry.peBlue = 255 -
ColorGradient(2).Value
' if modify system palette.
If Check1.Value Then
      NewPaletteEntry.peFlags = PC_RESERVED
Else
      NewPaletteEntry.peFlags = &H0
End If
```

3. ทำการแก้ไขค่าของสีและแฟลกตามที่กำหนด โดยในการแก้ไขถาดสีตรรกะผู้อ่านต้องใช้ฟังก์ชัน SetPaletteEntries ในการแก้ไขค่าของสี และทำการ realize ด้วยฟังก์ชัน RealizePalette แต่ถ้าหากเป็นการแก้ไขถาดสีระบบ ผู้อ่านต้องใช้ฟังก์ชัน AnimatePalette ในการแก้ไขถาดสี และวินโดวส์ก็จะจัดการแก้ไขสีในถาดสีตรรกะที่มีการอ้างอิงถึงทันที

```
' if modify system palette.
If Check1.Value Then
      AnimatePalette hCurrentPalette, SelectColorIdx,_
            1, NewPaletteEntry
Else
      hSystemPalette = GetStockObject(DEFAULT_PALETTE)
      hCurrentPalette = SelectPalette(Picture1.hDC,_
            hSystemPalette, False)
      Dummy = SetPaletteEntries(hCurrentPalette,_
            SelectColorIdx, _ 1, NewPaletteEntry)
      hSystemPalette = SelectPalette(Picture1.hDC,_
            hCurrentPalette, False)
      Dummy = RealizePalette(Picture1.hDC)
End If
```

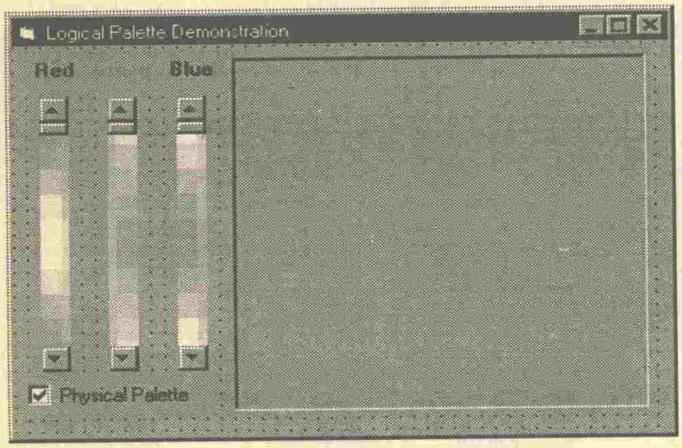
โปรแกรมตัวอย่าง LogPal

สำหรับโปรแกรมตัวอย่าง LogPal ประกอบด้วยเพียงฟอร์มเดียว โดยจะมี Picture1 สำหรับการใช้แสดงผลสีจากถาดสีตรรกะ และ Vertical Scroll Bar สำหรับใช้ในการปรับค่าของสีแดง (Red) สีเขียว (Green) และสีน้ำเงิน (Blue) ตามลำดับ โดยที่ Vertical Scroll Bar (คอนโทรล ColorGradient) ทั้งสาม ผู้เขียนได้กำหนดให้เป็นอะเรย์ (0 ถึง 2 เพื่อให้ง่ายในการใช้งาน ส่วนด้านล่างของฟอร์มก็จะมี Check Box สำหรับใช้เลือกถาดสีที่ต้องการแก้ไขว่าเป็นถาดสีตรรกะ (ไม่มีเครื่องหมายถูก) หรือถาดสีระบบ (มีเครื่องหมายถูก))

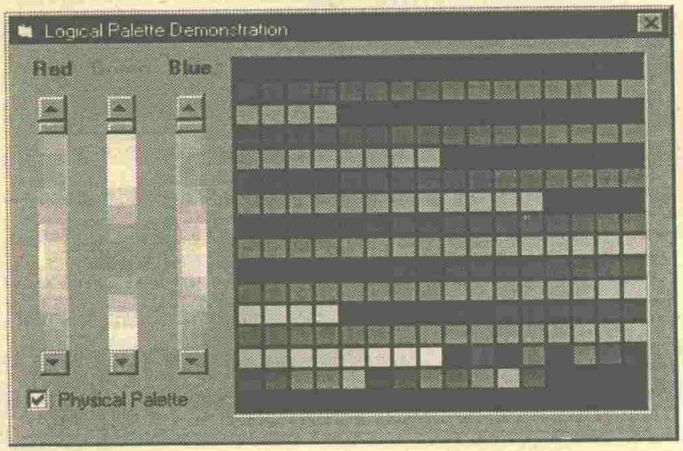
ส่วนการทำงานของโปรแกรมก็เริ่มจาก เมื่อผู้อ่านรันโปรแกรม รูทีนเหตุการณ์ Form_Load ก็จะถูกเรียก ซึ่งภายในรูทีนนี้ก็จะทำการกำหนดค่าของสีทั้ง 236 รายการที่ต้องการใช้ในการสร้างถาดสีตรรกะ โดยในการกำหนดค่าของสีนี้ผู้เขียนได้แยกออกเป็น 2 ส่วน ซึ่งในส่วนแรกก็จะเป็นการวนลูปสีละ 6 ครั้ง จำนวน 3 ลูป สำหรับสีแดง สีเขียว และสีน้ำเงินตามลำดับ ก็จะสามารถกำหนดค่าของสีได้ 216 รายการ ดังนี้

```
' create random logical color palette.
LogicalPalette.palVersion = &H300
LogicalPalette.palNumEntries = 236
For ClrR = 1 To 6
   For ClrG = 1 To 6
    For ClrB = 1 To 6
       ClrIdx = ((ClrR - 1) * 36) + ((ClrG - 1) * 6) + (ClrB - 1)
       LogicalPalette.palPalEntry(ClrIdx).peRed = ClrR *_
       (255 \ 6)
       LogicalPalette.palPalEntry(ClrIdx).peGreen = ClrG *_
       (255 \ 6)
       LogicalPalette.palPalEntry(ClrIdx).peBlue = ClrB *_
       255 \ 6)
       LogicalPalette.palPalEntry(ClrIdx).peFlags =_
       PC_RESERVED
    Next ClrB
   Next ClrG
 Next ClrR
```

ส่วนรายการลำดับที่ 217 ถึง 236 นั้นผู้เขียนก็ใช้วิธีการสุ่มค่าของสีขึ้นมาโดยการใช้ฟังก์ชัน RND ของ VB/Win ในการสุ่มตัวเลขระหว่าง 0 ถึง 1 ทั้งนี้เพื่อให้ผู้อ่านเห็นถึงความแตกต่างของสีที่เกิดจากการเรียงค่าแม่สี จากนั้นก็ทำการสร้างถาดสีตรรกะด้วยฟังก์ชัน CreatePalette



รูปที่ 2 ฟอร์ม Form1 ของโปรแกรม LogPal



รูปที่ 3 ฟอร์ม Form1 ของโปรแกรม LogPal ในขณะทำงาน

```
For ClrIdx = 217 To 236
  LogicalPalette.palPalEntry(ClrIdx).peRed = 255 * Rnd
  LogicalPalette.palPalEntry(ClrIdx).peGreen = 255 * Rnd
  LogicalPalette.palPalEntry(ClrIdx).peBlue = 255 * Rnd
  LogicalPalette.palPalEntry(ClrIdx).peFlags =_
      PC_RESERVED
Next ClrIdx
hCurrentPalette = CreatePalette(LogicalPalette)
SelectColorIdx = True
```

สำหรับโค้ดสุดท้ายของรูทีนเหตุการณ์นี้ ก็เป็นการย้ายฟอร์มมาแสดงผลที่กึ่งกลางจอภาพ ดังนี้

```
Me.Move (Screen.Width - Width) / 2, (Screen.Height - Height) / 2
```

จากนั้นเมื่อมีการแสดงผลฟอร์มรูทีนเหตุการณ์ Form_Paint ก็จะถูกเรียกทันที ซึ่งภายในรูทีนนี้ผู้เขียนก็ได้สร้างโค้ดสำหรับทำการแสดงผลสีทั้งหมดในถาดสีตรรกะลงในคอนโทรล Picture1 โดยเริ่มต้นด้วยการเลือกถาดสีด้วยฟังก์ชัน SelectPalette ลงใน device context ของคอนโทรล Picture1 จากนั้นจึงทำการ realize ด้วยฟังก์ชัน RealizePalette และทำการคำนวณขนาดของสี่เหลี่ยมที่ใช้ในการแสดงผลสีแต่ละสี โดยกำหนดให้แสดงสีแถวละ 16 สีรวมทั้งหมด 16 แถว ดังนี้

```
hSystemPalette = SelectPalette(Picture1.hDC,_
hCurrentPalette, False)
Dummy = RealizePalette(Picture1.hDC)
BoxW = Picture1.ScaleWidth / 16
BoxH = Picture1.ScaleHeight / 16
```

ต่อไปจึงทำการวาดสีทั้ง 256 สีลงในคอนโทรล Picture1 โดยการใช้แปรงทาสีที่มีสีตามลำดับที่ถูกกำหนด ซึ่งแปรงทาสีนี้จะถูกสร้างด้วยฟังก์ชัน CreateSolidBrush ซึ่งรายละเอียดในขั้นตอนนี้ผู้เขียนได้กล่าวแล้วในบทความตอนที่ผ่านมา

```
' redraw color grid.
For ColorIdx = 0 To 255
   Row = ColorIdx \ 16 + 1
   Col = ColorIdx Mod 16 + 1
   hBrush = CreateSolidBrush(&H1000000 Or ColorIdx)
   ' try to use this line for above one.
   'hBrush = CreateSolidBrush(&H0 Or ColorIdx)
   Dummy = SelectObject(Picture1.hDC, hBrush)
   Dummy = Rectangle(Picture1.hDC, (Col - 1) * BoxW, _
   (Row - 1) * BoxH, Col * BoxW, Row * BoxH)
   Dummy = SelectObject(Picture1.hDC,_
   GetStockObject(BLACK_BRUSH))
   Dummy = DeleteObject(hBrush)
Next ColorIdx
```

เมื่อทำการแสดงผลสีเสร็จแล้ว ก็จะมีการตรวจสอบว่ามีการเลือกสีหรือไม่ ซึ่งการเลือกสีก็ทำได้โดยการคลิ๊กเม้าส์ภายใน Picture1 เท่านั้น โดยถ้าหากมีการเลือกสีค่าของตัวแปร SelectColorIdx ก็จะมากกว่าหรือเท่ากับ 0 (มีค่าตั้งแต่ 0 ถึง 255) ก็ให้สร้างสี่เหลี่ยมที่มีขนาดโตกว่าสีเหลี่ยมที่ใช้แสดงสีปกติ 1.5 เท่า ด้วยวิธีเดียวกันทุกประการ ดังนี้

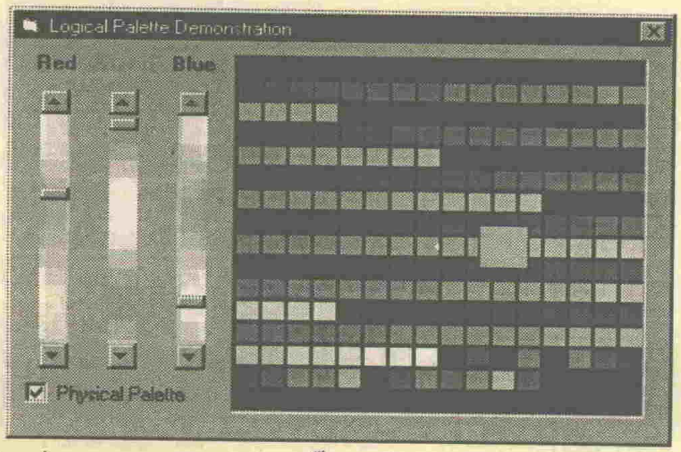
เปิดโลกมัลติมีเดียด้วย VB/Win
ตอน จัดการกับ Palette ของวินโดวส์ 95 ด้วย VB/Win #2

```
If SelectColorIdx >= 0 Then
 Row = SelectColorIdx \ 16 + 1
 Col = SelectColorIdx Mod 16 + 1
 hBrush = CreateSolidBrush(&H1000000 Or SelectColorIdx)
 Dummy = SelectObject(Picture1.hDC, hBrush)
 Dummy = Rectangle(Picture1.hDC, MaxInt((Col - 1.5) * BoxW,0),_
       MaxInt((Row - 1.5) * BoxH, 0), (Col + 0.5) * BoxW, _
       (Row + 0.5) * BoxH)
 Dummy = SelectObject(Picture1.hDC,etStockObject(BLACK_BRUSH))
 Dummy = DeleteObject(hBrush)
End If
```

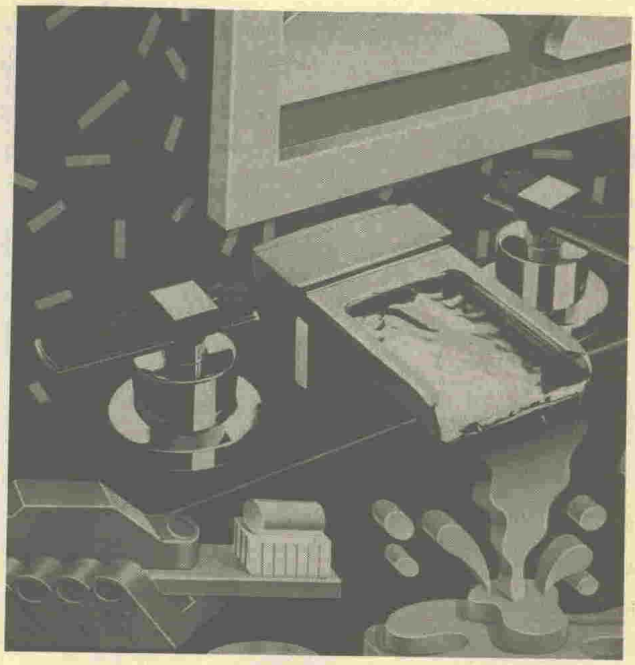
เมื่อโปรแกรมมีการสร้างตารางสีเสร็จแล้ว ก็ขอให้ผู้อ่านทดลองใช้เม้าส์คลิ๊กภายในบริเวณตารางสี ที่ลำดับสีใดก็ได้ ซึ่งเมื่อมีการคลิ๊กภายในบริเวณตารางสี ก็จะปรากฏสี่เหลี่ยมขนาด 1.5 เท่าของปกติปรากฏที่สีรายการนั้น ๆ ดังในรูปที่ 4 และแถบ vertical scroll bar ก็จะแสดงถึงค่าของสีแดง สีเขียว และสีน้ำเงิน ตามลำดับของสีรายการที่ถูกเลือก ซึ่งโค้ดที่ทำงานในส่วนนี้ก็ได้จากรูทีนเหตุการณ์ Picture1_MouseDown ซึ่งจะเริ่มต้นด้วยการคำนวณลำดับของสีที่ถูกเลือกดังนี้

```
    ' calculate color index.
    BoxW = Picture1.ScaleWidth / 16
    BoxH = Picture1.ScaleHeight / 16
    Row = (Y \ BoxH)
    Col = X \ BoxW
    SelectColorIdx = Row * 16 + Col
```

และอ่านค่าแม่สีทั้งสามจากลำดับสีที่ถูกเลือกด้วยฟังก์ชัน GetPaletteEntries โดยที่ข้อมูลที่อ่านก็จะถูกจัดเก็บลงในตัวแปร CurrentPaletteEntry ซึ่งเราก็จะนำค่าดังกล่าวมากำหนดให้กับคอนโทรล ColorGradient ทั้งสามและกำหนดให้ตัวแปร ChooseColorFlag มีค่าเท่ากับ False เพื่อให้การรีเฟรชคอนโทรล Picture1 ใหม่โดยรูทีนเหตุการณ์ Picture1_Paint



รูปที่ 4 แสดงการเลือกสีโดยการคลิ๊กเม้าส์ที่รายการที่ต้องการ



```
    ' retrive color value from palette.
    Dummy = GetPaletteEntries(hCurrentPalette,_
      SelectColorIdx, 1, CurrentPaletteEntry)

    ChooseColorFlag = True
    ColorGradient(0).Value = 255 - _
          CurrentPaletteEntry.peRed
    ColorGradient(1).Value = 255 - _
          CurrentPaletteEntry.peGreen
    ColorGradient(2).Value = 255 - _
          CurrentPaletteEntry.peBlue
    ChooseColorFlag = False
    Picture1_Paint
```

เมื่อผู้อ่านทำการเลือกสีแล้ว ผู้อ่านสามารถที่จะแก้ไขค่าของสีที่ลำดับที่ถูกเลือกได้โดยการเลื่อนแถบเลื่อนของคอนโทรล ColorGradient ทั้งสามตามค่าที่ต้องการ ซึ่งในขณะที่ผู้อ่านเลื่อนเคอร์เซอร์นั้น สีที่ปรากฏในสี่เหลี่ยมขนาด 1.5 เท่า ก็จะเปลี่ยนไปตามส่วนผสมของแม่สีทั้งสามโดยอัตโนมัติ ซึ่งในขณะที่มีการแก้ไขค่าของสีโดยการเลื่อนเคอร์เซอร์ของคอนโทรล ColorGradient นั้นรูทีนเหตุการณ์ ColorGradient_Scroll ก็จะถูกเรียก ซึ่งโค้ดภายในรูทีนเหตุการณ์นี้ก็จะทำการแก้ไขค่าของสีในถาดสีตรรกะหรือถาดสีระบบ ทั้งนี้ก็ขึ้นกับว่าผู้อ่านเลือกที่จะแก้ไขค่าของสีในถาดสีอะไร โดยการเลือกที่คอนโทรล check box ตรงด้านล่างของฟอร์ม โดยเลือกคอนโทรล check box ถ้าหากต้องการแก้ไขค่าของสีในถาดสีระบบ หรือไม่เลือกคอนโทรล check box ถ้าหากต้องการแก้ไขค่าของสีในถาดสีตรรกะ สำหรับความแตกต่างของการแก้ไขค่าสีในถาดสีทั้งสอง ก็ขอให้ผู้อ่านสังเกตจากการทดลองรันโปรแกรมก็แล้วกันครับ เพราะจะเห็นภาพมากกว่าครับ ซึ่งโค้ดในส่วนของรูทีนเหตุการณ์ ColorGradient_Scroll ผู้เขียนก็ได้อธิบายไปแล้วข้างต้น

เปิดโลกมัลติมีเดียด้วย VB/Win

ตอน จัดการกับ Palette ของวินโดวส์ 95 ด้วย VB/Win #2

ถึงจุดนี้นอกจากผู้อ่านจะสามารถแสดงผลถาดสีตรรกะได้แล้ว ก็ยังสามารถทำการแก้ไขค่าของสีในถาดสีตรรกะและถาดสีระบบได้อีกด้วย ซึ่งผู้อ่านก็คงจะสามารถนำไปประยุกต์ใช้งานได้ตามความเหมาะสม และสำหรับในบทความต่อหน้าก็จะเป็นการเจาะลึกลงไปอีกขั้นซึ่งจะเป็นการทำการหมุนลำดับของสีในถาดสี (color cycling technique) ซึ่งจะช่วยเสริมความสามารถของการใช้งานคอนโทรล Picture box ได้เป็นอย่างมากเลยทีเดียว เนื่องจาก VB/Win ไม่มีคำสั่งที่สนับสนุนงานด้านนี้เลย ดังนั้นผู้อ่านจึงต้องใช้ฟังก์ชันวินโดว์ API ในการทำงานดังกล่าว พบกันใหม่ในตอนต่อไปสวัสดีครับ

ลิสติงที่ 1
ไฟล์โปรเจ็กต์ LOGPAL.VBP

```
Form=LogPal.Frm
ProjWinSize=103,394,248,215
ProjWinShow=2
IconForm="LogPal"
HelpFile=""
Title="Logical Palette Demonstration"
ExeName32="LogPal.Exe"
Path32="c:\"
HelpContextID="0"
StartMode=0
VersionCompatible32="0"
MajorVer=1
MinorVer=0
RevisionVer=0
AutoIncrementVer=0
ServerSupportFiles=0
VersionComments="Distributed by Microcomputer Magazine _
                 Published by SE-ED Co.ltd.  "
VersionCompanyName="SE-ED Co.,Ltd."
VersionFileDescription="Logical Palette Demonstration"
VersionLegalCopyright="Mr.Sutthisak Phongthanapanich"
VersionProductName="Microcomputer Magazine"
CondComp="Win32 = -1"
```

ลิสติงที่ 2 ไฟล์ฟอร์ม LOGPAL.FRM

```
VERSION 4.00
Begin VB.Form LogPal
   BorderStyle     =   3  'Fixed Dialog
   Caption         =   "Logical Palette
Demonstration"
   ClientHeight    =   3765
   ClientLeft      =   1290
   ClientTop       =   1845
   ClientWidth     =   5760
   FillColor       =   &H8000000A&
   FillStyle       =   0  'Solid
   BeginProperty Font {0BE35203-8F91-11CE-9DE3-
00AA004BB851}
      Name              =   "MS Sans Serif"
      Size              =   8.25
      Charset           =   222
      Weight            =   700
      Underline         =   0   'False
      Italic            =   0   'False
      Strikethrough     =   0   'False
   EndProperty
   ForeColor       =   &H00000000&
   Height          =   4170
   Left            =   1230
   LinkTopic       =   "Form1"
   MaxButton       =   0   'False
   MinButton       =   0   'False
   ScaleHeight     =   251
   ScaleMode       =   3  'Pixel
   ScaleWidth      =   384
   ShowInTaskbar   =   0   'False
   Top             =   1500
   Width           =   5880
   Begin VB.CheckBox Check1
      Caption         =   "Physical Palette"
      BeginProperty Font {0BE35203-8F91-11CE-
9DE3-00AA004BB851}
         Name              =   "MS Sans Serif"
         Size              =   8.25
         Charset           =   222
         Weight            =   400
         Underline         =   0   'False
         Italic            =   0   'False
         Strikethrough     =   0   'False
      EndProperty
      Height          =   255
      Left            =   120
      TabIndex        =   7
      Top             =   3240
      Value           =   1  'Checked
      Width           =   1575
   End
   Begin VB.VScrollBar ColorGradient
      Height          =   2655
      Index           =   2
      LargeChange     =   5
      Left            =   1440
      Max             =   255
      TabIndex        =   3
      Top             =   480
      Width           =   255
   End
   Begin VB.VScrollBar ColorGradient
      Height          =   2655
      Index           =   1
      LargeChange     =   5
      Left            =   840
      Max             =   255
```

ลิสติงที่ 2 (ต่อ)

```
        TabIndex        =   2
        Top             =   480
        Width           =   255
     End
     Begin VB.VScrollBar ColorGradient
        Height          =   2655
        Index           =   0
        LargeChange     =   5
        Left            =   240
        Max             =   255
        TabIndex        =   1
        Top             =   480
        Width           =   255
     End
     Begin VB.PictureBox Picture1
        Height          =   3420
        Left            =   1920
        ScaleHeight     =   224
        ScaleMode       =   3  'Pixel
        ScaleWidth      =   239
        TabIndex        =   0
        Top             =   120
        Width           =   3645
     End
     Begin VB.Label Label3
        Alignment       =   2  'Center
        Caption         =   "Blue"
        ForeColor       =   &H00FF0000&
        Height          =   255
        Left            =   1320
        TabIndex        =   6
        Top             =   120
        Width           =   495
     End
     Begin VB.Label Label2
        Alignment       =   2  'Center
        Caption         =   "Green"
        ForeColor       =   &H0000FF00&
        Height          =   255
        Left            =   720
        TabIndex        =   5
        Top             =   120
        Width           =   495
     End
     Begin VB.Label Label1
        Alignment       =   2  'Center
        Caption         =   "Red"
        ForeColor       =   &H000000FF&
        Height          =   255
        Left            =   120
        TabIndex        =   4
        Top             =   120
        Width           =   495
     End
End
Attribute VB_Name = "LogPal"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
Option Explicit
'
' Project name  : LOGPAL.VBP
' Program name  : Logical Palette Demonstration
' VB/Win Module : LOGPAL.FRM
' DLL filename  : -
' Copyright (C)1997 by SUTTHISAK PHONGTHANAPANICH
' Distributed by: SE-ED Co.,Ltd.
'

Private Type PALETTEENTRY
    peRed       As Byte
    peGreen     As Byte
    peBlue      As Byte
    peFlags     As Byte
End Type
```

→

ลิสติงที่ 2 (ต่อ)

```
Private Type LOGPALETTE
    palVersion       As Integer
    palNumEntries    As Integer
    palPalEntry(255) As PALETTEENTRY
End Type

Private Const BLACK_BRUSH = 4
Private Const DEFAULT_PALETTE = 15
' palette index used for animation technique.
Private Const PC_RESERVED = &H1

Private Declare Function CreatePalette Lib "gdi32" _
        (lpLogPalette As LOGPALETTE) As Long
Private Declare Function CreateSolidBrush Lib "gdi32" _
        (ByVal crColor As Long) As Long
Private Declare Function DeleteObject Lib "gdi32" _
        (ByVal hObject As Long) As Long
Private Declare Function GetStockObject Lib "gdi32" _
        (ByVal nIndex As Long) As Long
Private Declare Function GetPaletteEntries Lib "gdi32" (ByVal hPalette _
        As Long, ByVal wStartIndex As Long, ByVal wNumEntries As _
        Long, lpPaletteEntries As PALETTEENTRY) As Long
Private Declare Function GetDeviceCaps Lib "gdi32" _
        (ByVal hDC As Long, ByVal nIndex As Long) As Long
Private Declare Function RealizePalette Lib "gdi32" _
        (ByVal hDC As Long) As Long
Private Declare Function Rectangle Lib "gdi32" _
        (ByVal hDC As Long, ByVal X1 As Long, ByVal Y1 As Long, _
        ByVal X2 As Long, ByVal Y2 As Long) As Long
Private Declare Function SelectObject Lib "gdi32" _
        (ByVal hDC As Long, ByVal hObject As Long) As Long
Private Declare Function SelectPalette Lib "gdi32" _
        (ByVal hDC As Long, ByVal hPalette As Long, _
        ByVal bForceBackground As Long) As Long
Private Declare Function SetPaletteEntries Lib "gdi32" _
        (ByVal hPalette As Long, ByVal wStartIndex As Long, _
        ByVal wNumEntries As Long,lpPaletteEntries As PALETTEENTRY) As Long

Private Declare Sub AnimatePalette Lib "gdi32" (ByVal hPalette _
        As Long, ByVal wStartIndex As Long, ByVal wNumEntries As _
        Long, lpPaletteColors As PALETTEENTRY)

Private ChooseColorFlag As Boolean
Private SelectColorIdx As Long
Private hSystemPalette As Long
Private hCurrentPalette As Long
Private PaletteEntries(256) As PALETTEENTRY

Private Dummy As Long

Private Sub ColorGradient_Change(Index As Integer)
    ColorGradient_Scroll Index
End Sub

Private Sub ColorGradient_Scroll(Index As Integer)
    Dim NewPaletteEntry As PALETTEENTRY

    ' modify color in logical palette or system palette.
    NewPaletteEntry.peRed = 255 - ColorGradient(0).Value
    NewPaletteEntry.peGreen = 255 - ColorGradient(1).Value
    NewPaletteEntry.peBlue = 255 - ColorGradient(2).Value
    If Check1.Value Then
        NewPaletteEntry.peFlags = PC_RESERVED
        If Not ChooseColorFlag Then
            AnimatePalette hCurrentPalette, SelectColorIdx, _
                           1, NewPaletteEntry
        End If
    Else
        NewPaletteEntry.peFlags = 0
        hSystemPalette = GetStockObject(DEFAULT_PALETTE)
        hCurrentPalette = SelectPalette(Picture1.hDC, hSystemPalette, False)
        Dummy = SetPaletteEntries(hCurrentPalette, SelectColorIdx, _
    1, NewPaletteEntry)
```

→

ลิสติงที่ 2 (ต่อ)

```
        hSystemPalette = SelectPalette(Picture1.hDC, hCurrentPalette, False)
        Dummy = RealizePalette(Picture1.hDC)
    End If
End Sub

Private Sub Form_Load()
    Dim LogicalPalette As LOGPALETTE
    Dim ClrIdx As Integer
    Dim ClrR As Integer, ClrG As Integer, ClrB As Integer

    Randomize Timer

    ' create random logical color palette.
    LogicalPalette.palVersion = &H300
    LogicalPalette.palNumEntries = 236
    For ClrR = 1 To 6
        For ClrG = 1 To 6
            For ClrB = 1 To 6
              ClrIdx = ((ClrR - 1) * 36) + ((ClrG - 1) * 6) + (ClrB - 1)
              LogicalPalette.palPalEntry(ClrIdx).peRed = ClrR * (255 \ 6)
              LogicalPalette.palPalEntry(ClrIdx).peGreen = ClrG * (255 \ 6)
              LogicalPalette.palPalEntry(ClrIdx).peBlue = ClrB * (255 \ 6)
              LogicalPalette.palPalEntry(ClrIdx).peFlags = PC_RESERVED
            Next ClrB
        Next ClrG
    Next ClrR
    For ClrIdx = 217 To 236
        LogicalPalette.palPalEntry(ClrIdx).peRed = 255 * Rnd
        LogicalPalette.palPalEntry(ClrIdx).peGreen = 255 * Rnd
        LogicalPalette.palPalEntry(ClrIdx).peBlue = 255 * Rnd
        LogicalPalette.palPalEntry(ClrIdx).peFlags = PC_RESERVED
    Next ClrIdx
    hCurrentPalette = CreatePalette(LogicalPalette)
    SelectColorIdx = True
End Sub

Private Sub Form_Unload(Cancel As Integer)
    hSystemPalette = GetStockObject(DEFAULT_PALETTE)
    Dummy = SelectPalette(Picture1.hDC, hSystemPalette, False)
    Dummy = DeleteObject(hCurrentPalette)
End Sub

Private Function MaxInt(A As Single, B As Single) As Integer
    MaxInt = IIf(A > B, A, B)
End Function

Private Sub Picture1_MouseDown(Button As Integer, _
 Shift As Integer, X As Single, Y As Single)
    Dim CurrentPaletteEntry As PALETTEENTRY
    Dim Row As Integer, Col As Integer
    Dim BoxH As Integer, BoxW As Integer

    ' calculate color index.
    BoxW = Picture1.ScaleWidth / 16
    BoxH = Picture1.ScaleHeight / 16
    Row = (Y \ BoxH)
    Col = X \ BoxW
    SelectColorIdx = Row * 16 + Col

    ' retrive color value from palette.
    Dummy = GetPaletteEntries(hCurrentPalette, SelectColorIdx, _
      1, CurrentPaletteEntry)
    ChooseColorFlag = True
    ColorGradient(0).Value = 255 - CurrentPaletteEntry.peRed
    ColorGradient(1).Value = 255 - CurrentPaletteEntry.peGreen
    ColorGradient(2).Value = 255 - CurrentPaletteEntry.peBlue
    ChooseColorFlag = False
    Picture1_Paint
End Sub

Private Sub Picture1_Paint()
    Dim Row As Integer, Col As Integer
    Dim BoxH As Integer, BoxW As Integer
    Dim ColorIdx As Integer
    Dim hBrush As Long                                          →
```

ลิสติงที่ 2 (ต่อ)

```
    hSystemPalette = SelectPalette(Picture1.hDC, hCurrentPalette, False)
    Dummy = RealizePalette(Picture1.hDC)
    BoxW = Picture1.ScaleWidth / 16
    BoxH = Picture1.ScaleHeight / 16

    ' redraw color grid.
    For ColorIdx = 0 To 255
        Row = ColorIdx \ 16 + 1
        Col = ColorIdx Mod 16 + 1
        hBrush = CreateSolidBrush(&H1000000 Or ColorIdx)
        Dummy = SelectObject(Picture1.hDC, hBrush)
        Dummy = Rectangle(Picture1.hDC, (Col - 1) * BoxW, _
                (Row - 1) * BoxH, Col * BoxW, Row * BoxH)
        Dummy = SelectObject(Picture1.hDC, GetStockObject(BLACK_BRUSH))
        Dummy = DeleteObject(hBrush)
    Next ColorIdx

    ' color selected, redraw focus color box.
    If SelectColorIdx >= 0 Then
        Row = SelectColorIdx \ 16 + 1
        Col = SelectColorIdx Mod 16 + 1
        hBrush = CreateSolidBrush(&H1000000 Or SelectColorIdx)
        Dummy = SelectObject(Picture1.hDC, hBrush)
        Dummy = Rectangle(Picture1.hDC, MaxInt((Col - 1.5) * BoxW, 0), _
                MaxInt((Row - 1.5) * BoxH, 0), (Col + 0.5) * BoxW, _
                (Row + 0.5) * BoxH)
        Dummy = SelectObject(Picture1.hDC, GetStockObject(BLACK_BRUSH))
        Dummy = DeleteObject(hBrush)
    End If
End Sub
```

■

เทคนิคประจำคอลัมน์

ถาม ในด้านมัลติมีเดีย CD มีอยู่ด้วยกันกี่ประเภท ?

ตอบ ในปัจจุบันมีอยู่ด้วยกัน 5 เทคโนโลยี ดังนี้

1. CD-DA (Compact Disc-Digital Audio)
   แผ่นซีดีที่จัดเก็บข้อมูลเสียงดิจิตอลคุณภาพสูง ซึ่งมีความจุสูงสุดเท่ากับ 73 นาที ซึ่งจะจัดเก็บข้อมูลเสียงดิจิตอลคุณภาพสูงในรูปแบบข้อมูลแสง (optical data) ซึ่งเราเรียกซีดีประเภทนี้ว่า "Red Book audio" ซึ่งเป็นมาตรฐานที่ใช้กันในปัจจุบัน
2. CD-I (Compact Disc Interactive)
   ตัวเล่นมัลติมีเดียแบบอินเทอร์แอคทีฟ ซึ่งสามารถอ่านข้อมูลแสงในรูปแบบพิเศษได้ (CD-I) โดยสามารถต่อเชื่อมกับเครื่องรับโทรทัศน์ได้ โดยที่มาตรฐาน CD-I นั้นจะสามารถจัดเก็บข้อมูลได้หลาย ๆ ชนิด เช่น อิมเมจภาพนิ่ง เสียง และวิดีโอภาพเคลื่อนไหว เราเรียกซีดีประเภทนี้ว่า "Green Book" ซึ่งมาตรฐาน CD-I นี้ได้รับการพัฒนาโดยบริษัทฟิลิปส์ (Phillips)
3. CD-ROM (Compact Disc-Read Only Memory)
   แผ่นซีดีที่จัดเก็บข้อมูลในรูปแบบข้อมูลแสง ซึ่งเป็นที่รู้จักกันดีในปัจจุบัน เพราะโปรแกรมส่วนใหญ่ในปัจจุบันจะถูกบันทึกลงในแผ่นซีดีรอมนี้เอง เราเรียกซีดีประเภทนี้ว่า "Yellow Book"
4. CDTV (Commodore Dynamic Total Vision)
   ตัวเล่นมัลติมีเดีย CD-ROM แบบอินเทอร์แอคทีฟ โดยสามารถต่อเชื่อมกับเครื่องรับโทรทัศน์ได้ ซึ่งเป็นเทคโนโลยีที่มุ่งเน้นเจาะตลาดในครัวเรือน และเป็นคู่แข่งที่สำคัญของเทคโนโลยี CD-I
5. CD-XA (CD-ROM eXtended Architecture)
   เป็นเทคโนโลยีที่ได้รับการพัฒนาขึดความสามารถเพิ่มเติมจากมาตรฐาน CD-ROM ที่มีการผสมผสานรูปแบบการจัดเก็บข้อมูลเสียงในรูปแบบบีบอัดข้อมูลไปพร้อม ๆ กับข้อมูลอื่น ๆ ในแผ่น CD-ROM (interleaved) ในรูปแบบที่เป็นอิสระกัน เช่นเดียวกับรูปแบบที่ใช้ใน CD-I แต่ CD-XA จะมีการกำหนดวิธีในการอ่านข้อมูลจากแผ่นซีดี โดยการผสมข้อมูลเท็กซ์และกราฟิกเข้าในแทร็กเดียวกัน ซึ่งในปัจจุบันมาตรฐานเพิ่มเติมนี้ได้ถูกรวมเอาไว้ในเทคโนโลยี CD-ROM "Yellow Book"